MS Appeal Brief-Patents
PATENT
0579-1112

IN THE U.S. PATENT AND TRADEMARK OFFICE BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re application of	Appeal No.
Jean-Bernard FISCHER et al.	Conf. 1488
Application No. 10/563,554	Group 2437
Filed April 14, 2006	Examiner Shewaye Gelagay

METHOD FOR MAKING SECURE THE EXECUTION OF A COMPUTER PROGRAM, IN PARTICULAR IN A SMART CARD

APPEAL BRIEF

Mail Stop Appeal Brief - Patents June 8, 2010
Assistant Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

MAY IT PLEASE YOUR HONORS:

(i) Real Party in Interest

The real party of interest in this appeal is Oberthur Card Systems SA, 102 Boulevard Malesherbes, 75017 Paris, France.

(ii) Related Appeals and Interferences

None.

(iii) **Status of Claims**

Claims 1-6, 9, 11 and 13-18 are pending and stand rejected. Claims 7, 8, 10, 12 were previously cancelled. This appeal is taken from the final rejection of claims 1-6, 9, 11 and 13-18.

(iv) **Status of Amendments**

The Amendment of April 9, 2010, amended claims 13 and 17 and canceled claims 7 and 8. The Advisory Action of April 30, 2010 did not indicate the status of the amended claims. In a subsequent phone call the Examiner indicated the claims were entered for purposes of appeal.

(v) **Summary of Claimed Subject Matter**

Independent claim 1 is directed to a real time distributed database system.

Claim 1 recites a method of making the execution of a computer program secure (page 1, lines 1-6), the method comprising a processor (Fig. 1, processor 110, page 24, lines 8-12) performing: a step of stacking a predetermined value in an instruction stack of the program (page 3, lines 21 and 32, XXX); said predetermined value being an address of an anomaly processing function (page 8, lines 8-10), during the normal execution of the program, a step of removing said predetermined value from the instruction stack, without executing the anomaly processing function (page 7, lines 28-35); and a step of unstacking said stack wherein if said predetermined value is unstacked, the anomaly processing function is executed (page 8, lines 11-17).

(vi) **Grounds of Rejection to be Reviewed on Appeal**

The first issue on appeal is whether claims 1, 2, 5, 6, 9, 11 and 13-15 are obvious, in the meaning of 35 USC §103(a), based on Stahl, U.S. Patent No. 5,274,817 in view of Szor, U.S. Patent Publication No. 2004/0158729, in view of Choi et al. A New Stack Buffer Overflow Hacking Defense Technique with Memory Address Confirmation, ICICS 2001, pages 146-159.

The second issue on appeal is whether claims 3 and 4 would have been obvious, in the meaning of 35 USC §103(a), based on Stahl, U.S. Patent No. 5,274,817 in view of Szor, U.S. Patent Publication No. 2004/0158729, in view of Choi et al. A New Stack Buffer Overflow Hacking Defense Technique with Memory Address Confirmation, ICICS 2001, pages 146-159 in view of McInerney, U.S. Patent No. 5,956,479.

The third issue on appeal is whether claim 16 would have been obvious, in the meaning of 35 USC §103(a), based on Stahl, U.S. Patent No. 5,274,817 in view of Szor, U.S. Patent Publication No. 2004/0158729, in view of Choi et al. A New Stack Buffer Overflow Hacking Defense Technique with Memory Address Confirmation, ICICS 2001, pages 146-159 in view of Zisowski, U.S. Patent Publication No. 2003/0188174.

The fourth issue on appeal is whether claims 17 and 18 would have been obvious, in the meaning of 35 USC §103(a), based on Stahl, U.S. Patent No. 5,274,817 in view of Szor, U.S. Patent Publication No. 2004/0158729, in view of Choi et al. A New Stack

Buffer Overflow Hacking Defense Technique with Memory Address Confirmation, ICICS 2001, pages 146-159 in view of Pritchard, U.S. Patent Publication No. 2002/0166067.

(vii) **Arguments**

(1) Arguments Concerning the First Ground of Rejection, Claims 1, 2, 5, 6, 9, 11 and 13-15 would not have been obvious based on Stahl, Szor and Choi.

Stahl discusses comparing the signature word stored on the stack with the subroutine entry address code, passing control to the return address if the compared values are equal and executing a software interrupt if the compared values are not equal.

Szor discusses terminating a call when it is determined that a call module is not in an executable portion of memory.

Choi discusses stack overflow prevention techniques for unknown attack detection.

Claim 1

On page 4 of the Office Action, the Office asserts that Stahl, col. 1, lines 62-67; col. 4, lines 57-64, col. 5, lines 8-17 disclose "a step of unstacking said stack wherein if said predetermined value is unstacked, the anomaly processing function is executed," as in claim 1.

In particular, the Examiner states on page 4 "*if the signature word stored on the stack matches the entry address of the subroutine which was just execute ...* ***if the compared values do not match****, it is assumed that an error has occurred and control is passed to the block where a software interrupt is* *executed*." (Emphasis added)

However, such a comparison is opposite of the claim wording where the anomaly function is executed "if said predetermined value is unstacked" (i.e. read from the stack). Said differently, according to the invention, the anomaly function is executed by reading the predetermined value in the stack; in this case, the value in the stack matches the predetermined value previously stored.

In the Office Action, the Office does not assert that any other reference teaches such a feature.

However, in the Advisory Action the Examiner states

> Choi teaches protecting systems against stack attacks by inserting a canary word to the stack just before, the return address when a function has been called, and when the function returns, ***StackGuard checks the canary word. If the canaray word has not been changed***, then the function progresses normally. Applicant's argument that Stahl teaches away is not persuasive Stahl and Choi as well as Applicant's invention relates to keeping the integrity of execution of a software specifically by keeping the integrity of the stack. ***Therfore, they are related to the same problem being solved secure execution of a program***. [Emphasis added]

Thus, it is noted that the Examiner seems to back off from the assertion made in the final Office Action in favor of asserting that Choi discloses the instant feature of claim 1. Further, the Office asserts that as the references are in the same field, that they are combinable.

However, if one is to accept, *arguendo*, that Choi discloses the feature of the claim, then Choi would teach a feature exactly opposite of that posed by Stahl. As such, the two references teach away from combination.

It is noted that the Examiner does not assert, nor have the Applicants found, that Szor discloses such a feature. Further, had such a feature been found, it would have been opposite the teachings of Stahl and therefore likewise teach away from combination.

For at least the reasons discussed above, claims 1 and the claims dependent therefrom are not obvious over the combination of Stahl, Szor and Choi.

(2) Arguments Concerning the Second Ground of Rejection, Claims 3 and 4 would not have been obvious based on Stahl, Szor, Choi and McInerney.

Claims 3 and 4

Claims 3 and 4 are dependent from claim 2 which in turn is dependent from claim 1. As the Applicants hold claim 1 is allowable, claims 3 and 4 are allowable as being dependent from an allowable base claim.

For at least the reasons discussed above, Stahl, Szor, Choi and McInerney, taken separately or in combination, fail to render obvious the features of claims 3 and 4.

(3) Arguments Concerning the Second Ground of Rejection, Claim 16 would not have been obvious based on Stahl, Szor, Choi and Zisowski.

Claim 16

Claim 16 dependent from claim 15 which in turn is dependent from claim 1. As the Applicants hold claim 1 is allowable, claim 16 allowable as being dependent from an allowable base claim.

For at least the reasons discussed above, Stahl, Szor, Choi and Zisowski, taken separately or in combination, fail to render obvious the features of claim 16.

(4) Arguments Concerning the Second Ground of Rejection, Claims 17 and 18 would not have been obvious based on Stahl, Szor, Choi and Pritchard.

Claims 17 and 18

Claim 17 is dependent from claim 16 which in turn is dependent from claim 15 which is in turn dependent from claim 1. Claim 18 is dependent from claim 1. As the Applicants hold claim 1 is allowable, claims 17 and 18 are allowable as being dependent from an allowable base claim.

For at least the reasons discussed above, Stahl, Szor, Choi and Pritchard, taken separately or in combination, fail to render obvious the features of claims 17 and 18.

Conclusion

Appellants respectfully urge that the rejections on appeal should not be maintained, and respectfully requests that these rejections be reversed.

The fee for the Appeal Brief in the amount of $540.00 is being paid online herewith by credit card.

If necessary, the Commissioner is hereby authorized in this, concurrent, and future submissions, to charge any underpayment or credit any overpayment to Deposit Account No. 25-0120 for any additional fees required under 37 C.F.R. § 1.16 or under 37 C.F.R. § 1.17.

Respectfully submitted,

YOUNG & THOMPSON

/James J. Livingston, Jr./

James J. Livingston, Jr.
Reg. No. 55,394
209 Madison Street, Suite 500
Alexandria, VA 22314
Telephone (703) 521-2297
Telefax (703) 685-0573
(703) 979-4709

JJL/jaa

June 9, 2010

(viii) Claims Appendix

1. A method of making the execution of a computer program secure, the method comprising:

a processor performing:

- a step of stacking a predetermined value in an instruction stack of the program; said predetermined value being an address of an anomaly processing function,

- during the normal execution of the program, a step of removing said predetermined value from the instruction stack, without executing the anomaly processing function; and

- a step of unstacking said stack wherein if said predetermined value is unstacked, the anomaly processing function is executed.

2. The method according to claim 1, wherein said stacking and unstacking steps are respectively associated with elements of at least one subset of instructions of said program.

3. The method according to claim 2, wherein said elements are respectively an opening bracket and a closing bracket in a system of brackets.

4. The method according to claim 2, wherein said unstacking step is associated with a return instruction of said program or a subroutine of said program.

5. The method according to claim 1, wherein said program is written in a programming language including at least one of a first instruction whose execution implements said stacking step and a second instruction whose execution implements said unstacking step.

6. The method according to claim 5, wherein the second instruction terminates said program or a subroutine of said program.

9. The method according to claim 1, wherein said program includes at least one call to a subroutine, wherein said stacking step is effected before said call and said predetermined value is eliminated from said stack during execution of said subroutine.

11. The method according to claim 1, wherein said programming includes at least one call to a subroutine, wherein said stacking step is effected during execution of said subroutine and said predetermined value is eliminated from said stack after execution of said subroutine.

13. A computer readable information recording medium with a computer program recorded thereon, said information recording medium totally or partially removable, in particular a CD-ROM, or a magnetic medium, comprising a hard disk or diskette, wherein it includes instructions of the computer program for implementing a method according to claim 1 when the computer program is loaded into and executed by an electronic data processing system.

14. A computer readable information recording medium with a computer program recorded thereon, said computer program including instructions for executing a method according to claim 1 when that program is loaded into and executed by an electronic data processing system.

15. An electronic entity that has been made secure wherein it includes means for implementing a method according to claim 1.

16. The electronic entity according to claim 15 wherein it is a smart card.

17. The electronic entity according to claim 16, wherein the anomaly processing function destroys an operating system of said smart card.

18. The method according to claim 1, wherein the anomaly processing function destroys an operating system of a smart card.

(ix) **Evidence Appendix**

None.

(x) Related Proceedings Appendix

None.